Image Generator

written by Philipp Klein
(PhilippKlein004 @ GitHub)

Version: 1.0
Languages: Java

TABLE OF CONTENTS

1. How to install the plugin

Before you install the plugin on your server, make sure that it is running on Minecraft version 1.8 to 1.8.8. Support for other versions is not guaranteed.

After setting up your server, simply place the JAR file into the "plugins" folder. Once the upload is complete, join the server and execute the command "/reload". This command instructs the server to reload all resources, including the plugins.

After receiving confirmation that the reload has finished, execute the command "/plugins" to view all installed plugins on your server. You should see the name "ImageGenerator" in green, indicating that the plugin has been successfully installed.

2. How to use the plugin

There are two essential commands, in the plugin:

1. “/generateimage [URL]”
   → This is the primary command for generating new images. The image will be rendered at the coordinates 0, 4, 0 (x, y, z). This plugin is specifically designed for super flat worlds! Please ensure that the URL provided is valid and that the host allows the use of their images.

2. “/clearimage”
   → Clears the previously drawn image from the world.

Note: It can occur that, if a large image is rendered, you’ll get kicked from the server due to a timeout. This is normal, and you should be able to rejoin right after you were kicked.

Quick Demonstration

Here’s an example: I took a cute picture of my budgie, and I uploaded it, to my website. The Link to the image:

https://kleinworkspace.com/images/Image_Generator/myBudgie.png

Before uploading the picture to my website, I resized it to make it smaller, ensuring faster rendering. Originally, the image was 3024 x 4032 pixels. Using Figma, I reduced its size to 315 x 420 pixels.

Since my server only has 2GB of RAM, I was careful not to overwhelm it right from the start with the first image. I wanted to ensure its stability and avoid causing any potential issues or crashes due to the limited memory.

Figure 1: My cute budgie (315 x 420)

With the URL copied in the clipboard the complete command looks like this:

/generateimage https://kleinworkspace.com/images/Image_Generator/myBudgie.png

After pressing enter, the plugin will download the image locally. If successful, a message will display the completed download as well the size of the image.

Figure 2: Communication with the user

When the image has been downloaded the rendering begins. You'll be informed if the image has been successfully rendered.

Depending on the size of the image, you can choose between flying up and viewing it from a greater distance or using scaled maps. Be aware that images in these maps become pixelated very quickly, so back up maps frequently. For the image of my budgie, I flew 170 blocks high and could see everything clearly.

Figure 3: The rendered image of my budgie

3. How the plugin works and how it was made

As stated earlier, the plugin downloads images from a given URL and renders them in-game. For the program to choose a block for a pixel, I used the weighted Euclidean distance. It has the following form (Source):

$$d(C1, C2) = \sqrt{0.3 \times (R1 - R2)^2 + 0.59 \times (G1 - G2)^2 + 0.11 \times (B1 - B2)^2}$$

The letters stand for the colors of each pixel ***R***ed, ***G***reen and ***B***lue. Before the distances can be calculated the blocks in Minecraft have to be assigned to a specific color code.

I've positioned myself approximately 70 blocks away and taken a screenshot of each block. Using the "Digital Color Meter" – a built-in app in macOS – I extracted the color for each block and noted it down. In total, I've examined 74 blocks and noted their Material, ID and color code.

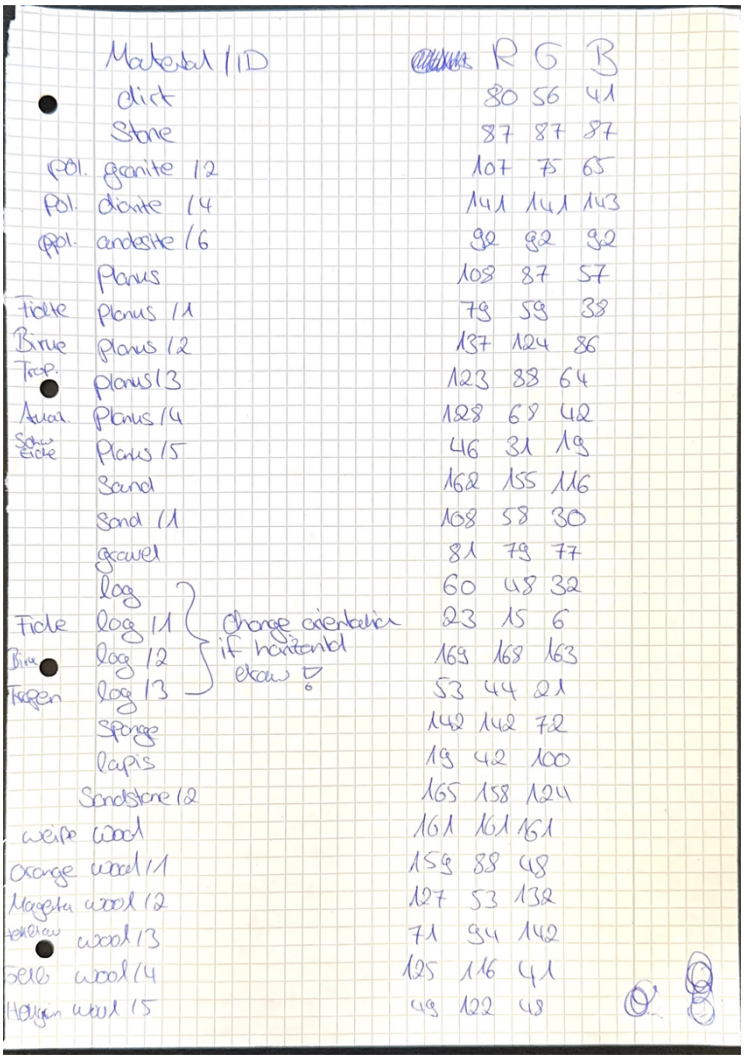

Figure 4: Excerpt of my notes for each blocks metadata and color code

The program goes through each pixel and calculates the distance to every block, that was noted. Here's an example:

We take a pixel of the nose of my budgie, which has the color code (40, 80, 116) according to the "Digital Color Meter".

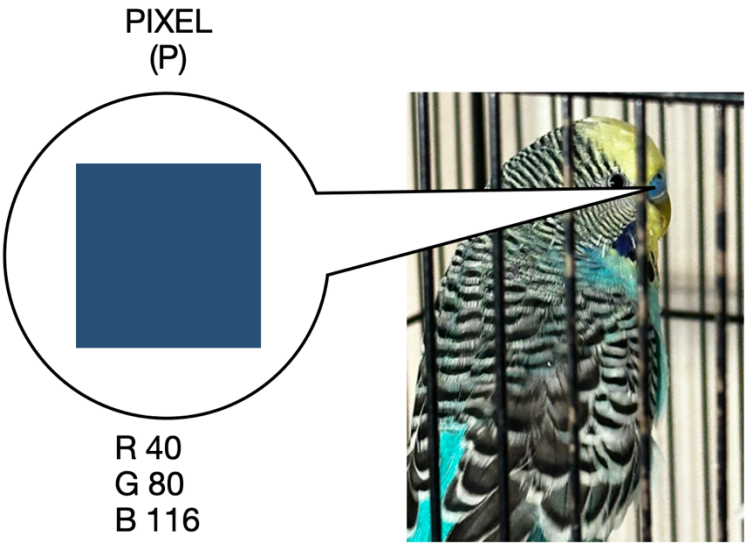

Figure 5: Pixel of my budgies nose

The following blocks are given with their color codes:

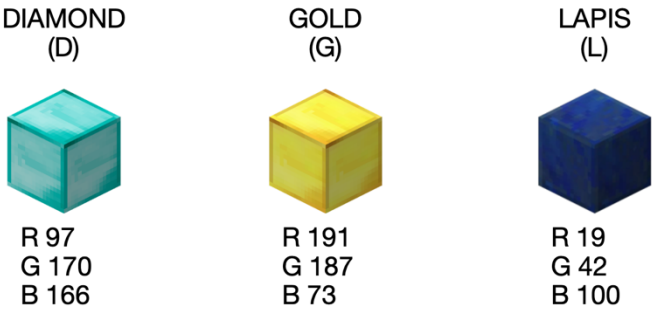

Figure 6: Blocks with their color codes

Now we can calculate the distances for each block to the pixel:

$$d(P,D) = \sqrt{0.3 \times (40-97)^2 + 0.59 \times (80-170)^2 + 0.11 \times (116-166)^2} \approx 77.644$$

$$d(P,G) = \sqrt{0.3 \times (40-191)^2 + 0.59 \times (80-187)^2 + 0.11 \times (116-73)^2} \approx 117.467$$

$$d(P,L) = \sqrt{0.3 \times (40-19)^2 + 0.59 \times (80-42)^2 + 0.11 \times (116-100)^2} \approx 31.818$$

As we can see, the lapis lazuli block has the shortest distance to the given pixel. If this remains the case for the remaining 71 blocks, then this block will be placed at the pixel's position.